Chapter 1

What Kind of Protection Do You Need?

COPYRIGHTED MATERIAL

The term *data protection* means different things to different people. Rather than asking what *kind* of protection you need, you should ask what data protection *problem* you are trying to solve. Security people discuss data protection in terms of *access*, where authentication, physical access, and firewalls are the main areas of focus. Other folks talk about protecting the integrity of the data with antivirus or antimalware functions. This chapter discusses protecting your data as an assurance of its availability in its current or previous forms.

Said another way, this book splits data protection into two concepts. We'll define *data protection* as preserving your data and *data availability* as ensuring the data is always accessible.

So, what are you solving for — protection or availability? The short answer is that while you'd like to say both, there is a primary and a secondary priority. More importantly, as we go through this book, you'll learn that it is almost never one technology that delivers both capabilities.

In the Beginning, There Were Disk and Tape

Disk was where data lived — always, we hoped.

Tape was where data rested — forever, we presumed.

Both beliefs were incorrect.

Because this book is focused on Windows data protection, we won't go back to the earliest days of IT and computers. But to appreciate where data protection and availability are today, we will briefly explore the methods that came before. It's a good way for us to frame most of the technology approaches that are available today. Understanding where they came from will help us appreciate what each is best designed to address.

We don't have to go back to the beginning of time for this explanation or even back to when computers became popular as mainframes. Instead, we'll go back to when Windows was first becoming a viable server platform.

During the late 1980s, local area networks (LANs) and servers were usually Novell NetWare. More notably for the readers of this book, data protection typically equated to connecting a tape drive to the network administrator's workstation. When the administrator went home at night, the software would log on as the administrator, presumably with full access rights, and protect all the data on the server.

In 1994, Windows NT started to become a server operating system of choice, or at least a serious contender in networking, with the grandiose dream of displacing NetWare in most environments.

Even with the "revolutionary" ability to connect a tape drive directly to your server, your two choices for data protection were still either highly available disk or nightly tape. With those as your only two choices, you didn't need to identify the difference between data protection and data availability. Data protection in those days was (as it is now) about preventing data loss from

happening, if possible. These two alternatives, highly available disk or nightly tape, provided two extremes where your data loss was measured at either zero or in numbers of days.

The concept of *data availability* was a misnomer. Your data either was available from disk or would hopefully be available if the restore completed, resulting in more a measure of restore reliability than an assurance of productive uptime. That being said, let's explore the two sides of today's alternatives: data availability and data protection.

Overview of Availability Mechanisms

Making something more highly available than whatever uptime is achievable by a standalone server with a default configuration sounds simple — and in some ways it is. It is certainly easier to engage resiliency mechanisms within and for server applications today than it was in the good ol' days. But we need to again ask the question "What are you solving for?" in terms of availability. If you are trying to make something more available, you must have a clear view of what might break so that something would be unavailable — and then mitigate against that kind of failure.

In application servers, there are several layers to the server — and any one of them can break (Figure 1.1).

FIGURE 1.1
Layers of a server

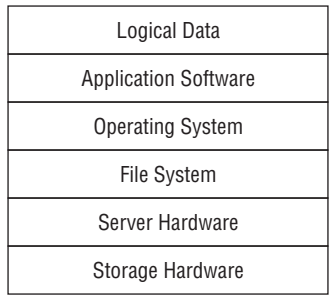

Figure 1.1 isn't a perfect picture of what can break within a server. It does not include the infrastructure — such as the network switches and routers between the server and the users' workstations. It doesn't include the users themselves. Both of these warrant large sections or books in their own right. In many IT organizations, there are server people, networking people, and desktop people. This book is for server people, so we will focus on the servers in the scenario and assume that our infrastructure is working and that our clients are well connected, patched, and knowledgeable, and are running applications compatible with our server.

For either data protection or data availability, we need to look at how it breaks — and then protect against it. Going from top to bottom:

- If the logical data breaks, it is no longer meaningful. This could be due to something as dire as a virus infection or an errant application writing zeros instead of ones. It could also be as innocent as the clicking of Save instead of Save As and overwriting your good data with an earlier draft. This is the domain of backup and restore — and I will cover that in the "Overview of Protection Mechanisms" section later in this chapter. So, for now, we'll take it off the list.
- In the software layers, if the application fails, then everything stops. The server has good data, but it isn't being served up to the users. Chapters 5 through 9 will look at a range of technologies that offer built-in availability. Similarly, if the application is running on an

operating system (OS) that fails, you get the same result. But it will be different technologies that keep the OS running rather than the application — and we'll delve deeply into both of these availability methods in Chapters 5 through 9.

- The file system is technically a logical representation of the physical zeros and ones on the disk, now presented as files. Some files are relevant by themselves (a text file), whereas other files are interdependent and only useful if accessed by a server application — such as a database file and its related transaction log files that make up a logical database within an application like MicrosoftSQL Server. The files themselves are important and unique, but in most cases, you can't just open up the data files directly. The server application must open them up, make them logically relevant, and offer them to the client software. Again, the file system is a good place for things to go badly and also an area where lots of availability technologies are being deployed. We'll look at these starting in Chapter 5.
- In the hardware layers, we see server and storage listed separately, under the assumption that in some cases the storage resides within the server and in other cases it is an appliance of some type. But the components will fail for different reasons, and we can address each of the two failure types in different ways. When we think of all the hardware components in a server, most electrical items can be categorized as either moving or static (no pun intended). The moving parts include most notably the disk drives, as well as the fans and power supplies. Almost everything else in the computer is simply electrical pathways. Because motion and friction wear out items faster than simply passing an electrical current, the moving parts often wear out first. The power supply stops converting current, the fan stops cooling the components, or the disk stops moving. Even within these moving components, the disk is often statistically the most common component to fail.

Now that we have one way of looking at the server, let's ask the question again: what are you concerned will fail? The answer determines where we need to look at availability technologies.

The easiest place to start is at the bottom — with storage.

Storage arrays are essentially large metal boxes full of disk drives and power supplies, plus the connecting components and controllers. And as we discussed earlier, the two types of components on a computer most likely to fail are the disk drives and power supplies. So it always seems ironic to me that in order to mitigate server outages by deploying mirrored storage arrays, you are essentially investing in very expensive boxes that contain several of the two most common components of a server that are most prone to fail. But because of the relatively short life of those components in comparison to the rest of the server, using multiple disks in a RAID-style configuration is often considered a requirement for most storage solutions.

Storage Availability

In the earlier days of computing, it was considered common knowledge that servers most often failed due to hardware, caused by the moving parts of the computer such as the disks, power supplies, and fans. Because of this, the two earliest protection options were based on mitigating hardware failure (disk) and recovering complete servers (tape). But as PC-based servers matured and standardized and while operating systems evolved and expanded, we saw a shift from hardware-level failures to software-based outages often (and in many early cases, predominantly) related to hardware drivers within the OS. Throughout the shift that occurred in the early and mid-1990s, general-purpose server hardware became inherently more reliable. However, it forced us to change how we looked at mitigating server issues because no matter how much redundancy we included and how many dollars we spent on mitigating hardware type outages, we were addressing only a

diminishing percentage of why servers failed. The growing majority of server outages were due to software — meaning not only the software-based hardware drivers, but also the applications and the OS itself. It is because of the shift in why servers were failing that data protection and availability had to evolve.

So, let's start by looking at what we can do to protect those hardware elements that can cause a server failure or data loss. In such cases, when a tier-one server vendor is respected in the datacenter space, I tend to dismiss the server hardware at first glance as the likely point of failure. So, storage is where we should look first.

Introducing RAID

No book on data protection would be complete in its first discussions on *disk* without summarizing what RAID is. Depending on when you first heard of RAID, it has been both:

- Redundant Array of *Inexpensive* Disks
- Redundant Array of *Independent* Disks

In Chapter 3, we will take an in-depth look at storage resiliency, including RAID models, but for now, the key idea is that statistically, the most common physical component of a computer to fail is a hard drive. Because of this, the concept of strapping multiple disks together in various ways (with the assumption that multiple hard drives will not all likely break at once) is now standard practice. RAID comes in multiple configurations, depending on how the redundancy is achieved or the disks are aligned:

Mirroring — RAID 1 The first thing we can do is to remove the single spindle (another term for a single physical disk, referring to the axis that all the physical platters within the disk spin on). In its simplest resolution, we mirror one disk or spindle with another. With this, the disk blocks are paired up so that when disk block number 234 is being written to the first disk, block number 234 on the second disk is receiving the same instruction at the same time. This completely removes a single spindle from being the single point of failure (SPOF), but it does so by consuming twice as much disk (which equates to at least twice the costs) power, cooling, and space within the server.

RAID 5, 1+0/10, and Others Chapter 3 will take us through all of the various RAID levels and their pros and cons, but, for now, the chief takeaway is that you are still solving a spindle-level failure. The difference between straight mirroring (RAID 1) and all other RAID variants is that you are not in a 1:1 ratio of production disk and redundant disk. Instead, in classic RAID 5, you might be spanning four disks where, for every N-1 (3 in this case) blocks being written, three of the disks get data and the fourth disk calculates parity for the other three. If any single spindle fails, the other three have the ability to reconstitute what was on the fourth, both in production on the fly (though performance is degraded) and in reconstituting a new fourth disk.

But it is all within the same array, storage cabinet, or shelf for the same server. What if your fancy RAID 5 disk array cabinet fails, due to two disks failing in a short timeframe, or the power failing, or whatever?

In principle, mirroring (also known as RAID-1) and most of the other RAID topologies are all attempts to keep a single hard drive failure from affecting the production server. Whether the strategy is applied at the hardware layer or within the OS, the result is that two or more disk drives act together to improve performance and/or mitigate outages. In large enterprises,

synchronously mirrored storage arrays provide even higher performance as well as resiliency. In this case the entire storage cabinet, including low-level controllers, power supplies, and hard drives, are all duplicated, and the two arrays mirror each other, usually in a synchronous manner where both arrays receive the data at exactly the same time. The production servers are not aware of the duplicated arrays and can therefore equally access either autonomous storage solution.

So far, this sounds pretty good. But there are still some challenges, though far fewer challenges than there used to be. Back then, disk arrays were inordinately more expensive than local storage. Add to that the cost and complexity of storage area network (SAN) fabrics and the proprietary adapters for the server(s), and the entire solution became cost-prohibitive for most environments. In 2002, Gartner's "Study of IT Trends" suggested that only 0.4 percent of all IT environments could afford the purchase price of synchronously mirrored storage arrays. For the other 99.6 percent, the cost of the solution was higher than the cost of the problem (potential data loss). Of course, that study is now eight years old. The cost of synchronously mirrored storage has gone down and the dependence on data has gone up, so it is likely that 0.4 percent is now too low of a number, but it is still a slim minority of IT environments. We will discuss this statistic, including how to calculate its applicability to you, as well as many metrics and decision points in Chapter 2.

While you could argue that the parity bits in a RAID configuration are about preserving the integrity of data, the bigger picture says that mirroring/striping technologies are fundamentally about protecting against a component-level failure — namely the hard drive. The big picture is about ensuring that the storage layer continuously provides its bits to the server, OS, and application. At the disk layer, it is always one logical copy of the blocks — regardless of how it is stored on the various spindles.

This concept gets a little less clear when we look at asynchronous replication, where the data doesn't always exactly match. But in principle, disk (hardware or array)-based "data protection" is about "availability."

Decision Question: Is It Really Mission Critical?

The first decision point, when looking at what kinds of data protection and availability to use, is whether or not the particular platform you are considering protecting is mission critical (we're ignoring cost factors until Chapter 2). But in principle, if you absolutely cannot afford to lose even a single mail message, database transaction, or other granular item of data, then a particular server or platform really is mission critical and you'll want to first look at synchronous storage as part of your solution along with a complementary availability technology for the other layers of the server (for example, application or OS).

Note that crossing the line between synchronous and asynchronous should be looked at objectively on a per-server or per-platform basis — instead of just presuming that everything needs the same level of protection.

Even for key workloads, the idea that they are mission critical and therefore immediately require synchronously mirrored disks and other extraordinary measures may not be universally justified. Consider two of the most common application workloads — SQL Server and Microsoft Exchange.

- In a large corporation with multiple Exchange Servers, you might find that the Exchange Server and/or the storage group that services email for the shipping department might be considered noncritical. As such, it may be relegated two nightly or weekly tape backups only. In that same company, the executive management team might require that their email

be assured 24/7 availability, including access on premises or from any Internet location. Even within one company, and for a single application, the protection method will differ. As an interesting twist, if the company we are discussing is Amazon.com, where their entire business is driven by shipping, that might be the most mission-critical department of all. Microsoft Exchange provides four different protection methods even within itself, not including array mirroring or disk- and tape-based backups (more on that in Chapter 7).

- Similarly, Microsoft SQL Server might be pervasive across the entire range of servers in the environment — but not every database may warrant mirroring, clustering, or replication at all.

If the data protection landscape was a graph, the horizontal X axis could be defined as a data loss, starting at 0 in the left corner and extending into seconds, minutes, hours, and days as we move across the graph. In short, what is your recovery point objective (RPO)?

We'll cover RPO and cost in Chapter 2. For now, know that RPO is one of the four universal metrics that we can use to compare the entire range of data protection solutions. Simply stated, RPO asks the question, "How much data can you afford to lose?"

In our rhetorical question, the key verb is *afford*. It is not *want* — nobody wants to lose any data. If cost was not a factor, it is likely that we would all unanimously choose zero data loss as our RPO. The point here is to recognize that even for your mission-critical, or let's just say most important, platforms, do you really need synchronous data protection — or would asynchronous be sufficient?

Should You Solve Your Availability Need with Synchronously Replicated Storage?

The answer is that "it depends." Here is what it depends on: If a particular server absolutely, positively cannot afford any loss of data, then an investment in synchronously mirrored storage arrays is a must. With redundancy within the spindles, along with two arrays mirroring each other, and redundant SAN fabric for the connectors, as well as duplicated host bus adapters (HBAs) within the server to the fabric, you can eliminate every SPOF in your storage solution. More importantly, it is the only choice that can potentially guarantee zero data loss.

This is our first decision question to identify what kinds of availability solutions we should consider:

- If we really need "zero data loss," we need synchronously mirrored storage (and additional layers of protection too).
- If we can tolerate anywhere from seconds to minutes of lost data, several additional technologies become choices for us — usually at a fraction of the cost.

Synchronous vs. Asynchronous

Synchronous versus asynchronous has been a point of debate ever since disk mirroring became available. In pragmatic terms, the choice to replicate synchronously or asynchronously is as simple as calculating the cost of the data compared with the cost of the solution. We will discuss this topic more in Chapter 2, as it relates to RPO and return on investment (ROI), but the short version is that if the asynchronous solution most appropriate for your workload protects data every 15 minutes, then what is 15 minutes' worth of data worth?

If the overall business impact of losing those 15 minutes' worth of data (including both lost information and lost productivity) is more expensive to the business than the cost of a mirrored and synchronous solution, then that particular server and its data should be synchronously mirrored at the storage level. As I mentioned earlier, the vast majority of corporate environments cannot justify the significantly increased cost of protecting those last (up to) 15 minutes of lost data — and therefore need an asynchronous protection model.

If your RPO truly and legitimately is zero, synchronously mirrored arrays are the only data protection option for you, or at least for that particular application, on that particular server, for that particular group within your company. To paraphrase a popular US television commercial tagline: "For everything else, there's asynchronous."

Asynchronous Replication

Even in environments where one platform demands truly zero data loss and therefore synchronous storage, the likelihood is that the remaining platforms in the same company do not. Again, the statistics will vary, but recall the extremes described in the previous sections: 0.4 percent of IT environments can cost-justify synchronously mirrored storage but only 1 percent of environments can rationalize half a business day of data loss with typically 1.5 days of downtime. If those statistics describe both ends of the data protection spectrum, then 98.6 percent of IT environments need a different type of data protection and/or availability that is somewhere in between the extremes. In short, while the percentages have likely changed and though your statistics may vary, most IT environments need better protection than nightly tape, which is less expensive than synchronous arrays.

In the Windows space, starting around 1997, the delivery of several asynchronous solutions spawned a new category of data protection and availability software, which delivered host-based (running from the server, not the array) replication that was asynchronous.

Asynchronous replication, by design, is a disk-to-disk replication solution between Windows servers. It can be done throughout the entire day, instead of nightly, which addresses the mainstream customer need of protecting data more frequently than each night. To reduce costs, asynchronous replication software reduces cost in two different dimensions:

Reduced Telecommunications Costs Synchronous mirroring assures zero data loss by writing to both arrays in parallel. The good news is that both of the arrays will have the same data. The bad news is that the servers and the applications could have a delay while both disk transactions are queued through the fabric and committed to each array. As distance increases, the amount of time for the remote disk to perform the write and then acknowledge it increases as well. Because a disk write operation is not considered complete until both halves of the mirror have acted on it, the higher-layer OS and application functions must wait for the disk operation to be completed on both halves of the mirror. This is inconsequential when the two arrays are side by side and next to the server. However, as the arrays are moved further from the server as well as from each other, the latency increases because the higher-layer functions of the server are waiting on the split disks. This latency can hinder the production application performance. Because of this, when arrays are geographically separated, companies must pay significant telecommunications costs to reduce latency between the arrays. In contrast to that, asynchronous replication allows for the primary disk on the production server to be written to at full speed, whereas the secondary disk has a replication target and is allowed to be delayed. As long as that latency is acceptable from a data loss perspective, one can be several minutes apart between the two disks and the result is appreciably reduced telecommunications costs.

Hardware Costs Typically, storage arrays that are capable of replication (synchronous or asynchronous) are appreciably more expensive than traditional disk chassis. Often, while the arrays are capable, they require separately licensed software to enable the mirroring or replication itself. As an alternative, replication can also be done within the server as an application-based capability, which is referred to as *host-based replication*. Host-based replication is done from server to server instead of array to array. As such, it is very typical to use less expensive hardware for the target server along with lower-performing drives for the redundant data. We will explore this topic later in Chapter 3.

Real World Scenario

The Platform and the Ecosystem

Years before I joined Microsoft, I was listening to a Microsoft executive explain one aspect of a partner ecosystem for large software developers (Microsoft in this case, but equally applicable to any OS or large application vendor). He explained that for any given operating system or application, there's always a long list of features and capabilities that the development team and product planners would like to deliver. Inevitably, if any software company decided to wait until every feature that they wanted was included in the product and it was well tested, then no software product would ever ship.

Instead, one of the aspects of the ecosystem of software developers is that those companies typically identify holes in the product that have enough customer demand to be profitable if developed. Thus, while Windows Server was still initially delivering and perfecting clustering, and while applications like SQL Server and Microsoft Exchange learned to live within a cluster, there was a need for higher availability and data protection that could be filled by third-party software, as discussed earlier.

The Microsoft speaker went on to explain that the reality of which holes in a product would be filled by the next version was based on market demand. This creates an unusual cooperative environment between the original developer and its partner ecosystem. Depending on customer demand, that need might be solved by the third-party vendor for one to three OS/application releases. But eventually, the hole will be filled by the original manufacturer — either by acquiring one of the third parties providing a solution or developing the feature internally. Either way, it allows all mainstream users of the OS/application to gain the benefit of whatever hole or feature previously filled by the third-party offering, because it was now built-in to the OS or application itself.

The nature and the challenge within the partner ecosystem then becomes the ability to recognize when those needs are being adequately addressed within the original Microsoft product to identify new areas of innovation that customers are looking for and build those.

Adding my data protection and availability commentary on that person's perspective — for nearly ten years, third-party asynchronous replication technologies were uniquely meeting the needs of Microsoft customers for data protection and availability by filling the gap between the previous alternatives of synchronous disk and nightly tape.

But as the largest application servers (SQL and Exchange) and Windows Server itself have added protection and availability technologies to meet those same customer needs within the most common scenarios of file services, databases, and email, the need for third-party replication for those workloads has significantly diminished. The nature of the ecosystem therefore suggests that third parties should be looking for other applications to be protected and made highly available, or identify completely different business problems to solve.

Undeniably, asynchronous host-based replication solved a real problem for Windows administrators for nearly 10 years. In fact, it solved two problems:

- Data *protection* in the sense that data could be "protected" (replicated) out of the production server more often than nightly, which is where tape is limited
- Data *availability* in the sense that the secondary copy/server could be rapidly leveraged if the primary copy/server failed

Asynchronous replication addressed a wide majority of customers who wanted to better protect their data, rather than making nightly tape backups, but who could not afford to implement synchronous storage arrays. We will cover asynchronous replication later in this book. For now, note that as a file system–based mechanism, asynchronous replication on its own is a category of data protection that is arguably diminishing as the next two technologies begin to flourish: Clustering and Asynchronous Replication.

Clustering

Ignoring the third-party asynchronous replication technologies for a moment, if you were a Microsoft expert looking at data protection in the early days of Windows Server, your only choice for higher availability was redundancy in the hardware through network interface card (NIC) teaming, redundant power supplies and fans, and of course, synchronous storage arrays. When the synchronous arrays are used for availability purposes, we must remember that hardware resiliency only addresses a small percentage of why a server fails. For the majority of server and service outages that were software based, Microsoft originally addressed this with Microsoft Cluster Services (MSCS) and other technologies that we'll cover later in this book.

MSCS originally became available well after the initial release of Windows NT 4.0, almost like an add-on or more specifically as a premium release with additional functionality. During the early days of Windows clustering, it was not uncommon for an expert-level Microsoft MCSE or deployment engineer (who might be thought of as brilliant with Windows in general) to struggle with some of the complexities in failover clustering. These initial challenges with clustering were exacerbated by the first generation of Windows applications that were intended to run on clusters, including SQL Server 4.21 and Exchange Server 5.0. Unfortunately, clustering of the applications was even more daunting.

In response to these challenges with the first built-in high availability mechanisms, many of the replication software products released in the mid-1990s included not only data protection but also availability. Initially, and some still to this day, those third-party replication technologies are burdened by support challenges based on how they accomplish the availability. But in principle, they work by either extending the Microsoft clustering services across sites and appreciable distances but allowing the cluster application to handle the failover. Or they use a proprietary method of artificially adding the failed server's name, IP, shares, and even applications to the replication target and then resuming operation. The industry leader in asynchronous replication is Double-Take from Double-Take Software, formerly known as NSI Software. Another example of this technology is WANSync from Computer Associates, acquired from XOsoft. XOsoft provided the initial WANSync for Data Protection, and followed up with WANSync^{HA}, which included data availability. We will discuss these products in Chapter 3.

MSCS continued to evolve and improve through Windows 2000, Windows Server 2003, and Windows Server 2003 R2. That trend of continued improvement would continue through the more recent Windows Server 2008 and the newly released Windows Server 2008 R2. But that isn't the whole story. MSCS will be covered in Chapter 6.

More and more, we see MSCS used for those applications that cannot provide availability themselves, or as an internal component or plumbing for their own built-in availability solutions, as opposed to an availability platform in its own right. Examples include Exchange cluster continuous replication (CCR) and database availability groups (DAGs), both of which we cover in Chapter 7.

Application Built-in Availability

From 1997 to 2005, asynchronous replication was uniquely filling the void for both data protection and data availability within many Windows Server environments — and as we discussed, Windows Server was not yet becoming commonplace except in larger enterprises with high IT professional skill sets. But while the clustering was becoming easier for those applications that could be clustered, another evolution was also taking place within the applications themselves. Starting around 2005, Microsoft began filling those availability and protection holes by providing native replication and availability within the products themselves.

File Services' Distributed File Services (DFS)

As the most common role that Windows Server is deployed into today, it should come as no surprise that the simple file shares role that enables everything from user home directories to team collaboration areas is a crucial role that demands high availability and data protection. To this end, Windows Server 2003 R2 released a significantly improved Distributed File System (DFS). DFS replication (DFS-R) provides partial-file synchronization up to every 15 minutes, while DFS namespace (DFS-N) provides a logical and abstracted view of your servers. Used in parallel, DFS-N transparently redirects users from one copy of their data to another, which has been previously synchronized by DFS-R. DFS is covered in Chapter 5.

SQL Server Mirroring

SQL Server introduced database mirroring with SQL Server 2005 and enhanced it in SQL Server 2008. Prior to this, SQL Server offered log shipping as a way to replicate data from one SQL Server to another. Database mirroring provides not only near-continuous replication but failover as well. And unlike the third-party approaches, database mirroring is part of SQL Server, so there are no supportability issues; in fact, database mirroring has a significantly higher performance than most third-party replication technologies because of how it works directly with the SQL logs and database mechanisms. By using a mirror-aware client, end users can be transparently and automatically connected to the other mirrored data, often within only a few seconds. SQL Server database protection will be covered in Chapter 8.

Exchange Replication

Exchange Server delivered several protection and availability solutions in Exchange Server 2007 and later in its first service pack. These capabilities essentially replicate data changes similarly to how SQL performs database mirroring, but leverages MSCS to facilitate failover. Exchange 2010 changed the capabilities again. The versions of Exchange availability solutions are as follows:

SCC Single copy cluster, essentially MSCS of Exchange, sharing one disk

LCR Local continuous replication within one server, to protect against disk-level failure

CCR Cluster continuous replication, for high availability (HA)

SCR Standby continuous replication, for disaster recovery (DR)

DAG Database availability group, for HA and DR combined

Exchange Server protection options will be covered in Chapter 7.

Decision Question: How Asynchronous?

Because built-in availability solutions usually replicate (asynchronously), we need to ask ourselves, "How asynchronous can we go?"

Asynchronous Is Not Synonymous with "Near Real Time" — It Means Not Synchronous

Within the wide spectrum of the replication/mirroring/synchronization technologies of data protection, the key variance is RPO. Even within the high availability category, RPO will vary from potentially zero to perhaps up to 1 hour. This is due to different vendor offerings within the space, and also because of the nature of asynchronous protection.

Asynchronous replication can yield zero data loss, if nothing is changing at the moment of failure. For replication technologies that are reactive (meaning that every time production data is changed, the replication technology immediately or at best possible speed transmits a copy of those changes), the RPO can usually be measured within seconds. It is not assured to be zero, though it could be if nothing had changed during the few seconds prior to the production server failure. For the same class of replication technologies, the RPO could yield several minutes of data loss if a significant amount of new data had changed immediately prior to the outage. This scenario is surprisingly common for production application servers that may choke and fail during large data imports or other high-change-rate situations, such as data mining or month-end processing.

However, not all solutions that deliver asynchronous replication for the purpose of availability attempt to replicate data in near real time. One good example is the DFS included with Windows Server (covered in Chapter 5). By design, DFS-R replicates data changes every 15 minutes. This is because DFS does not reactively replicate. In the earlier example, replication is immediately triggered because of a data change. With DFS-R, replication is a scheduled event. And with the recognition that the difference in user files likely does not have the financial impact necessitating replication more often than every 15 minutes, this is a logical RPO based on this workload.

Even for the commonplace workload of file serving, one solution does not fit all. For example, if you were using DRS-R not for file serving but for distribution purposes, it might be more reasonable to configure replication to occur only after hours. This strategy would still take advantage of the data-moving function of DFS-R, but because the end goal is not availability, a less frequent replication schedule is perfectly reasonable. By understanding the business application of how often data is copied, replicated, or synchronized, we can assess what kinds of frequency, and therefore which technology options, should be considered. We will take a closer look at establishing those quantifiable goals and assessing the technology alternatives in Chapter 2.

Availability vs. Protection

No matter how frequently you are replicating, mirroring, or synchronizing your data from the disk, host, or application level, the real question comes down to this:

> *Do you need to be able to immediately leverage the redundant data from where it is being stored, in the case of a failed production server or site?*

- If you are planning on resuming production from the replicated data, you are solving for *availability* and you should first look at the technology types that we've already covered (and will explore in depth in Chapters 5–9).
- If you need to recover to previous points in time, you are solving for *protection* and should first look at the next technologies we explore, as well as check out the in-depth guidance in Chapters 3 and 4.

We will put the technologies back together for a holistic view of your datacenter in Chapters 10–12.

Overview of Protection Mechanisms

Availability is part of the process of keeping the current data accessible to the users through

- Redundant storage and hardware
- Resilient operating systems
- Replicated file systems and applications

But what about yesterday's data? Or even this morning's data? Or last year's data? Most IT folks will automatically consider the word *backup* as a synonym for *data protection*. And for this book, that is only partially true.

Backup Backup implies nightly protection of data to tape. Note that there is a media type and frequency that is specific to that term.

Data Protection Data protection, not including the availability mechanisms discussed in the last section, still covers much more, because tape is not implied, nor is the frequency of only once per night.

Let's Talk Tape

Regardless of whether the tape drive was attached to the administrators' workstation or to the server itself, tape backup has not fundamentally changed in the last 15 years. It runs every night after users go home and is hopefully done by morning. Because most environments have more data than can be protected during their nightly tape backup window, most administrators are forced to do a full backup every weekend along with incremental or differentials each evening in order to catch up.

For the record, most environments would likely do full backups every night if time and money were not factors. Full backups are more efficient when doing restores because you can use a single tape (or tapes if needed) to restore everything. Instead, most restore efforts must first begin with restoring the latest full backup and then layer on each nightly incremental or latest differential to get back to the last known good backup.

Full, Incremental, and Differential Backups

We will cover backup to tape in much more detail as a method in Chapter 3, and in practice within System Center Data Protection Manager in Chapter 4, as one example of a modern backup solution. But to keep our definitions straight:

Full Backup Copies every file from the production data set, whether or not it has been recently updated. Then, additional processes mark that data as backed up, such as resetting the archive bit for normal files, or perhaps checkpointing or other maintenance operations within a transactional database. Traditionally, a full backup might be done each weekend.

Incremental Backup Copies only those files that have been updated since the last full or incremental backup. Afterward, incremental backups do similar postbackup markups as done by full backups, so that the next incremental will pick up where the last one left off. Traditionally, an incremental backup might be done each evening to capture only those files that changed during that day.

Differential Backup Copies only those files that have been updated since the last full backup. Differential backups do not do any postbackup processes or markups, so all subsequent differentials will also include what was protected in previous differentials until a full backup resets the cycle. Traditionally, differential backup might be done each evening, capturing more and more data each day until the next weekend's full backup.

NOTE If your environment only relies on nightly tape backup, then your company is agreeing to half a day of data loss and typically at least one and a half days of downtime per data recovery effort.

Let's assume that you are successfully getting a good nightly backup every evening, and a server dies the next day. If the server failed at the beginning of the day, you have lost relatively little data. If a server fails at the end of the day, you've lost an entire business day's worth of data. Averaging this out, we should assume that a server will always fail at the midpoint of the day, and since your last backup was yesterday evening, your company should plan to lose half of a business day's worth of data.

That is the optimistic view. Anyone who deals in data protection and recovery should be able to channel their pessimistic side and will recall that tape media is not always considered reliable. Different analysts and industry experts may place tape recovery failure rates at anywhere between 10 percent and 40 percent. My personal experience is 30 percent tape failure rate during larger recoveries, particularly when a backup job spans multiple physical tapes.

Let's assume that it is Thursday afternoon, and your production server has a hard drive failure. After you have repaired the hardware, you begin to do a tape restore of the data and find that one of the tapes is bad. Now you have three possible outcomes:

- If the tape that failed is last night's *differential*, where a differential backup is everything that has been changed since the last full backup, then you've only lost one additional day's worth of data. Last night's tape is no good, and you'll be restoring from the evening prior.
- If the tape that failed is an *incremental*, then your restorable data loss is only valid up until the incremental before this one. Let's break that down:
  - If you are restoring up to Thursday afternoon, your plan is to first restore the weekend's full backup, then Monday's incremental, then Tuesday's incremental, and then finally Wednesday's incremental.

- If it is Wednesday's incremental that failed, you can reliably restore through Tuesday night, and will have only lost one additional day's worth of data.
- But if the bad tape is Tuesday's incremental that failed, you can only reliably recover back to Monday night. Though you do have a tape for Wednesday, it would be suspect. And if you are unlucky, the data that you need was on Tuesday night's tape.

- The worst-case scenario, though, is when the full backup tape has errors. Now all of your incremental and differentials throughout the week are essentially invalid, because their intent was to update you from the full backup — which is not restorable. At this point, you'll restore from the weekend *before* that full backup. You'll then layer on the incrementals or differentials through last Thursday evening. In our example, as you'll recall, we said it was Thursday afternoon. When this restore process is finished, you'll have data from Thursday evening a week ago. You'll have lost an entire week of data. But wait, it gets worse. Remember, incrementals or differentials tend to automatically overwrite each week. This means that Wednesday night's backup job will likely overwrite last Wednesday's tape. If that is your rotation scheme, then your Monday, Tuesday, and Wednesday tapes are invalid because its full backup had the error. But after you restore the full backup of the weekend before, the days since then may have been overwritten. Hopefully, the Thursday evening of last week was a differential, not an incremental, which means that it holds all the data since the weekend prior and you'll still have lost only one week of data. If they were incrementals, you'll have lost nearly two weeks of data.

Your Recovery Goals Should Dictate Your Backup Methods

The series of dire scenarios I just listed is not a sequence of events, nor is it a calamity of errors. They all result from *one bad tape* and how it might affect your recovery goal, based on what you chose for your tape rotation.

One of the foundational messages you should take away from this book is that you should be choosing your backup methods and evaluating the product offerings within that category, based on how or what you want to recover.

This is not how most people work today. Most people protect their data using the best way that they know about or can believe that they can afford, and their backup method dictates their recovery scenarios.

Disk vs. Tape

The decision to protect data using disk rather than tape is another of the quintessential debates that has been around for as long as both choices have been viable. But we should not start the discussion by asking whether you should use disk or tape. As in the previous examples, the decision should be based on the question, "What is your recovery goal?"

More specifically, ask some questions like these:

- Will I usually restore selected data objects or complete servers?
- How frequently will I need to restore data?
- How old is the data that I'm typically restoring?

Asking yourself these kinds of questions can help steer you toward whether your recovery goals are better met with disk-based or tape-based technologies. Disk is not always better. Tape is not dead. There is not an all-purpose and undeniably best choice for data protection any more than there is an all-purpose and undeniably best choice for which operating system you should run on your desktop. In the latter example, factors such as which applications you will run on it, what peripherals you will attach to it, and what your peers use might come into play. For our purposes, data granularity, maximum age, and size of restoration are equally valid determinants.

We will cover those considerations and other specifics related to disk versus tape versus cloud in Chapter 3, but for now the key takeaway is to *plan how you want to recover, not how you want to be protected*.

As an example, think about how you travel. When you decide to go on a trip, you likely decide where you want to go before you decide how to get there.

If how you will recover your data is based on how you back up, it is like deciding that you'll vacation based on where the road ends — literally, jumping in the car and seeing where the road takes you. Maybe that approach is fine for a free-spirited vacationer, but not for an IT strategy. For me, I am not extremely free spirited by nature, so this does not sound wise for a vacation — and it sounds even worse as a plan for recovering corporate data after crisis. In my family, we choose what kind of vacation we want and then we decide how to get there. That is how your data protection and availability should be determined.

Instead of planning what kinds of recoveries you will do because of how you back up to nightly tape, turn that thinking around. Plan what kinds of recoveries you want to do (activities) and how often you want to do them (scheduling). This strategy is kind of like planning a vacation. Once you know what you want to accomplish, it is much easier to do what you will *need* to do so that you can do what you *want* to do.

Recovery is the goal. Backup is just the tax in advance that you pay so that you can recover the way that you want to. Once you have that in mind, you will likely find that tape-based backup alone is not good enough. It's why disk-based protection often makes sense — and almost always should be considered in addition to tape, not instead of tape.

Microsoft Improvements for Windows Backups

When looking at traditional tape backup, it is fair to say that the need was typically filled by third-party backup software. We discussed the inherent need for this throughout the chapter, and Windows Server has always included some level of a built-in utility to provide single-server and often ad hoc backups. From the beginning of Windows NT through Windows Server 2003 R2, Microsoft was essentially operating under an unspoken mantra of "If we build it, someone else will back it up." But for reasons that we will discuss in Chapter 4, that wasn't good enough for many environments. Instead, another layer of protection was needed to fill the gap between asynchronous replication and nightly tape backup.

In 2007, Microsoft released System Center Data Protection Manager (DPM) 2007. Eighteen months earlier, DPM 2006 had been released to address centralized backup of branch office data in a disk-to-disk manner prior to third-party tape backup. DPM 2007 delivered disk-to-disk replication, as well as tape backup, for most of the core Windows applications, including Windows Server, SQL Server, Exchange Server, SharePoint, and Microsoft virtualization hosts. The third generation of Microsoft's backup solution (DPM 2010) was released at about the same time as the printing of this book. DPM will be covered in Chapter 4.

Similar to how built-in availability technologies address an appreciable part of what asynchronous replication and failover were providing, Microsoft's release of a full-fledged backup product (in addition to the overhauled backup utility that is included with Windows Server) changes the ecosystem dynamic regarding backup. Here are a few of the benefits that DPM delivers compared to traditional nightly tape backup vendors:

- A single and unified agent is installed on each production server, rather than requiring separate modules and licensing for each and every agent's type, such as a SQL Server agent, open file handler, or a tape library module.
- Disk and tape are integrated within one solution, instead of a disk-to-disk replication from one vendor or technology patch together with a nightly tape backup solution built from a different code base.
- DPM 2010 is designed and optimized exclusively for Windows workloads, instead of a broad set of applications and OSs to protect, using a generic architecture. This is aimed at delivering better backups and the most supportable and reliable restore scenarios available for those Microsoft applications and servers.

The delivery by Microsoft of its own backup product and its discussion in this book is not to suggest that DPM is absolutely and unequivocally the very best backup solution for every single Windows customer in any scenario. DPM certainly has its strengths (and weaknesses) when compared with alternative backup solutions for protecting Windows. But underlying DPM, within the Windows operating system itself, are some internal and crucial mechanisms called Volume Shadow Copy Services (VSS). VSS, which is also covered in Chapter 4, is genuine innovation by Microsoft that can enable any backup vendor, DPM included, to do better backups by integrating closer to the applications and workloads themselves. Putting this back within the context of our data protection landscape: while we see more choices of protection and availability through third-party replication and built-in availability solutions, we are also seeing a higher quality and flexibility of backups and more reliability for restores through new mechanisms like VSS and DPM, which we will cover in Chapters 3 and 4.

Summary

In this chapter, you saw the wide variety of data protection and availability choices, with synchronous disk and nightly tape as the extremes and a great deal of innovation happening in between. Moreover, what was once a void between synchronously mirrored disks and nightly tape has been filled first by a combination of availability and protection suites of third-party products, and is now being addressed within the applications and the OS platforms themselves. The spectrum or landscape of data protection and availability technologies can be broken down into a range of categories shown in Figure 1.2.

FIGURE 1.2
The landscape of data protection and availability

Each of these capabilities will be covered in future chapters — including in-depth discussions on how they work as well as practical, step-by-step instructions on getting started with each of those technologies.

Selecting a data protection plan from among the multiple choices and then reliably implementing your plan in a cohesive way is critical — no matter how large or small, physical or virtual, your particular "enterprise" happens to be. There are a few key points that I hope you take away from this chapter:

- Start with a vision of what you want to recover, and then choose your protection technologies (usually plural) — not the other way around.
- Tape is not evil and disk is not perfect — but use each according to what each medium is best suited for.
- Be clear among your stakeholders as to whether you are seeking better protection or better availability. It's not always both and rarely does one technology or product cover them equally.
- Deliver "availability" within the workload/server if possible and achieve "protection" from a unified solution.
- No single protection or availability technology will cover you. Each addresses certain scenarios and you will want to look at a "balanced diet" across your enterprise — protecting each according to their needs.

Now that you know what you want to accomplish, let's move on to Chapter 2, where you'll learn how to quantify your solution, compare choices, and cost-justify.